QUARKUS

Java Rest Services Security

JEE Microservices

@ CGS IT – 2023

Version 1.0.5

Inhalt

- Rest Standards
- Rest Methoden
- JaxRs Library
- Java Beispiele
- Erweiterte Dokumentation
- Dokumentations Links

Quarkus - Security

- Das Beispiel basiert auf der Quarkus Basic Security Dokumentation
- Basic HTTP Security wird aktiviert
- JaxRS Security Annotations für Rest Services werden konfiguriert und verwendet

```
<!-- add quarkus security via jpa user entity -->
<dependency>
  <groupId>io.quarkus</groupId>
  <artifactId>quarkus-security-jpa</artifactId>
</dependency>
```

```
<!-- panachache quarkus jpa extensions -->
<dependency>
   <groupId>io.quarkus</groupId>
   <artifactId>quarkus-hibernate-orm-panache</artifactId>
</dependency>
```

CGS ITSolutions

Quarkus – Beispiel Secure API

Endpoint	Description
`/api/public`	The `/api/public` endpoint can be accessed anonymously.
`/api/admin`	The `/api/admin` endpoint is protected with role-based access control (RBAC), and only users who have been granted the `admin` role can access it. At this endpoint, the `@RolesAllowed` annotation enforces the access constraint declaratively.
`/api/users/me`	The `/api/users/me` endpoint is protected by RBAC. Only users that have the `user` role can access the endpoint. This endpoint returns the caller's username as a string.

 # Security

quarkus.http.auth.basic=true

CGS ITSolutions

Quarkus – Public Ressource

- Eine öffentliche Ressource wird mittels @PermitAll gekennzeichnet.
- Sie benötigt kein Login

```
import jakarta.annotation.security.PermitAll;


@Path("/api/public")
public class PublicResource {

    @GET
    @PermitAll
    @Produces(MediaType.TEXT_PLAIN)
    public String publicResource() {
        return "public";
    }
}
```

Quarkus - Roles Allowed - User Access

- Die Annotation @RolesAllowed kennzeichnet einen Resource Pfad als geschützt. Der Benutzer benötigt eine gültige User Session.
- Der Benutzer muss in der richtigen User-Rolle sein.

```
@Path("/api/users")
public class UserResource {

   @GET
   @RolesAllowed("user")
   @Path("/me")
   public String me(@Context SecurityContext securityContext)
{
      return securityContext.getUserPrincipal().getName();
   }
}
```

Quarkus – Security – Admin Role

- Der Admin User ist nur eine spezielle Rolle.
- Alle Benutzer (User) die diese Rolle zugwiesen haben, dürfen die Methode auch aufrufen

```
@Path("/api/admin")
public class AdminResource {

    @GET
    @RolesAllowed("admin")
    @Produces(MediaType.TEXT_PLAIN)
    public String adminResource() {
        return "admin";
    }
}
```

CGS ITSolutions

Quarkus – Panacache User

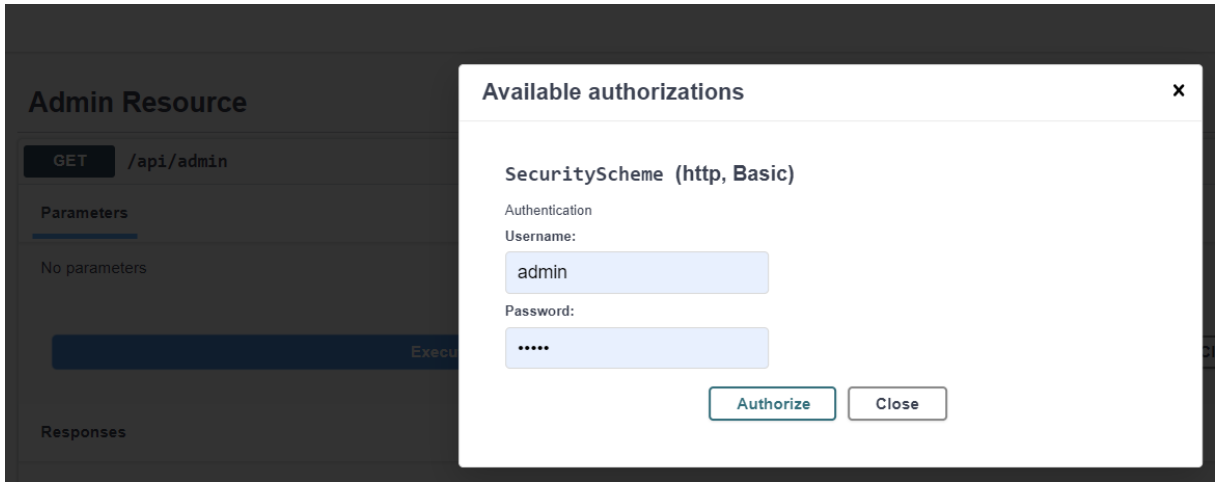
- Die Quarkus JPA Security Extension bietet Annotations zur Abbildung der user und Berechtigungen in der Datenbank an.
- Zusätzlich konfiguriert diese Extension das Quarkus System und Resteasy so, dass diese Benutzer auch an das JaxRS System entsprechend angebunden und richtig konfiguriert werden.
- Das Beispiel nutzt zusätzlich Quarkus Panache für ein einfacheres Datenbank Mapping.

```
import io.quarkus.hibernate.orm.panache.PanacheEntity;
import io.quarkus.security.jpa.UserDefinition;
import io.quarkus.security.jpa.Username;
import io.quarkus.security.jpa.Password;
import io.quarkus.security.jpa.Roles;

@Entity
@Table(name = "test_user")
@UserDefinition
public class User extends PanacheEntity {
   @Username
   public String username;
   @Password
   public String password;

   // einfache variante für rollen ohne extra tabelle
   @Roles
   public String role;
```

CGS ITSolutions

Quarkus – Swagger Login

- Swagger stellt für die Basic Security auch eine Login Maske zur Verfügung



QuarkusTest Support für Security

- Quarkus Test unterstüzt das Testen der Security mittels Rest Assured fluent API .basic

```
@Test
void shouldNotAccessUserWhenAdminAuthenticated() {
   given()
          .auth().preemptive()
          .basic("admin", "admin")
          .when()
          .get("/api/users/me")
          .then()
          .statusCode(HttpStatus.SC_FORBIDDEN);
}
```

CGS ITSolutions

QuarkusTest Security via Swagger & Curl

- Basic Security wird via Authorization Header an den Server übermittelt

```
curl -X 'GET' \
'http://localhost:8080/api/admin‘ \
 -H 'accept: text/plain‘ \
-H 'Authorization: Basic YWRtaW46YWRtaW4='
```

Danke für Ihre Aufmerksamkeit

CGS ITSolutions